

AN FPGA BOARD USED FOR DIGITAL LOGIC LABS

LEE, SUNGGU

Department of Electronic and Electrical Engineering., Pohang University of Science and Technology, Pohang 790-784, S. Korea

1. INTRODUCTION

Digital logic design is an essential part of the curriculum of a university's electrical engineering department. With the increasing importance of digital devices in today's society and the increasing complexity of the digital devices used, most electrical engineering departments offer more than one course on digital logic design. The digital logic courses offered typically consist of an introductory digital logic course followed by one or more courses focusing on aspects of advanced digital logic design. Such advanced digital logic design courses involve the use of hardware description languages, field-programmable logic arrays (FPGAs), and electronic design automation (EDA) computer-aided design (CAD) tools, including back-end tools such as Synopsis' logic synthesis tool [1], and front-end tools, such as Exsedia's Nimbus [2], used for graphical high-level design.

An essential part of an advanced digital logic design course is the lab, which should provide the students with the opportunity to practice designing complex digital logic circuits of various types using commercial software and hardware tools. In order to view the results of their design efforts and to experience the practical difficulties of actual hardware design, it is also essential for the students to be able to implement their circuits in hardware. In recent years, with increasing advances in FPGA technology, it is now feasible to use FPGAs to implement fairly large student projects such as 32-bit microprocessor designs.

A practical problem that arises when attempting to use FPGAs in a class lab is the availability of an FPGA board that can be used to prototype student designs. If sufficient money is available, a viable option is to purchase an FPGA prototyping board, such as the XESS board offered for use with Xilinx FPGAs [3]. However, such boards are costly and lag behind the state-of-the-art FPGA models offered. Also, if this type of "ready-to-use" FPGA prototyping solution is used in a university class, what is to distinguish the class from a short-term (e.g., one day to two weeks) industrial seminar? A university class should focus on the underlying theory, as well as the practical implementation aspects, involved with the use of tools and devices such as FPGAs. Thus, it would be more desirable if we could simply issue state-of-the-art FPGA chips to the students and have them create their own FPGA prototyping boards, using knowledge about how to connect to and configure the FPGAs based on information provided in data sheets.

A compromise solution is to design and fabricate a set of custom-designed printed circuit boards with state-of-the-art FPGA chips, and then use these FPGA boards to prototype student designs. Such a compromise solution may be necessary because state-of-the-art FPGA chips currently come in packages with over 100 pins in closely packed configurations; manual soldering of such chips is impractical, and the fabrication of printed circuit boards for individual FPGA designs is costly and time-consuming. Even with pre-made FPGA

prototyping boards, the students should still learn the entire FPGA configuration process, including the configuration file download procedure, and use this knowledge to use the FPGA prototyping board properly. The FPGA prototyping board should simply provide the wire connections required to enable configuration using externally provided header pins. This paper presents the design and philosophy behind one such custom-designed FPGA prototyping board. Based on the method and design presented in this paper, it is also possible to design other FPGA prototyping boards for other types of FPGA chips.

2. FPGA CONFIGURATION

An FPGA (field-programmable gate array), as its name implies, can be programmed in the field (i.e., by the end user). The overall FPGA configuration (also referred to as programming) process is as follows. First, the user must create his/her design using software EDA CAD tools (such as Exsedia's Nimbus) and then convert it into a "configuration" file format. Then the configuration file must somehow be downloaded to the FPGA chip. Control circuitry within the FPGA then takes this data and uses it to create or break connections at predetermined locations (typically horizontal-to-vertical crosspoints) dispersed throughout the chip. The circuitry within an FPGA chip is typically laid out in a rectangular two-dimensional matrix of configurable logic blocks (cells) separated by intervening rows and columns of wires (the routing channels). By creating customized connections, cells can be configured (or programmed) to implement specific subcircuits (such as 2-bit adders or counters) and connections can be made to the wires in the routing channels to connect cells in different locations. FPGAs can be classified into one-time-programmable devices, which can only be configured once in the field (these are typically non-volatile, i.e., circuit information is retained even when power is removed), and multiple-time-programmable devices, which can be repeatedly erased and configured (these are typically volatile, and thus lose their circuit information when power is removed).

As an example of an FPGA and the FPGA configuration process, let us examine the Spartan II XC2S200 FPGA produced by Xilinx. This is a 2.5 volt CMOS device with the equivalent of 200,000 logic gates, a 56-Kbit RAM memory, and 140-284 user I/O pins. As with most Xilinx FPGAs, it uses SRAM cells to store the configuration information, and is thus a volatile device which can undergo an almost unlimited number of reprogramming cycles. This chip supports the slave serial, master serial, slave parallel, and boundary scan modes of configuration, with the specific configuration mode selected by setting the values of three mode pins M0-M2. All configuration modes supported are synchronous modes, in which a separate clock signal (CCLK) is used to indicate the presence of valid configuration data on the data input (DIN) line. In the slave serial (or parallel) mode, the CCLK and DIN (or D0-D7) signals are generated externally and used to download the configuration data to the FPGA. In the master serial mode, the FPGA generates the CCLK signal and reads in the DIN signal at the appropriate times. Finally, the boundary scan mode "reuses" the I/O pins provided for the production testing of this device, referred to as the JTAG access port, in order to download configuration data in a slave serial-like manner.

3. DESIGN OF THE FPGA PROTOTYPING BOARD

The FPGA prototyping board was designed to facilitate experimentation with numerous different types of digital logic circuit designs. Since it was not designed to be a commercial product, software support was not provided. Instead, we simply relied on the configuration software available with most FPGA tools to download the configuration data to our board.

Figure 1 shows a block diagram of the FPGA prototyping board that was produced. There are several notable points about this board design. First, all user I/O pins are brought out to header pins so that they can be probed easily using logic analyzer or digital oscilloscope probes. If we wish to connect this board to another circuit board and drive inputs to the FPGA from the external board, that can be easily accomplished by connecting to the header pins provided.

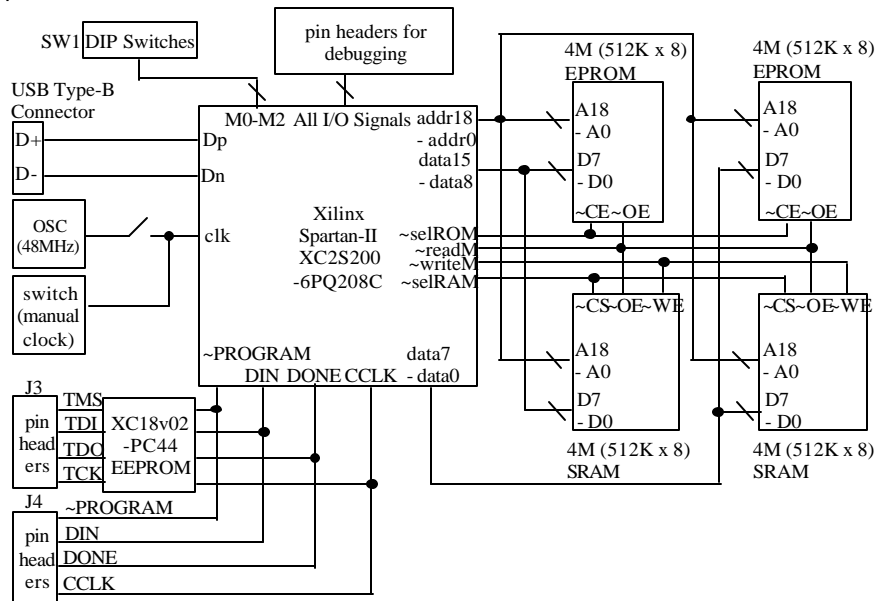


Fig. 1. Block diagram of a custom-designed FPGA prototyping board for a Xilinx Spartan II FPGA

As can be seen from Figure 1, connections have been provided for configuration using both slave serial mode and boundary scan mode. The specific mode used can be selected by connecting the mode pins (M0 – M2) to ground or Vdd. In particular, the boundary scan configuration pins have been connected to a serial EEPROM. Thus, configuration data can first be downloaded to the serial EEPROM using boundary scan mode. Then, since the EEPROM is a non-volatile device, power can be removed, the mode changed to master serial mode, and the configuration data downloaded to the FPGA at any time after the power is restored. The FPGA can also be configured directly using slave serial mode.

Since we wished to be able to experiment with modern I/O interfaces and memory devices, we also added connections to a USB (universal serial bus) port and connections to SRAM and EPROM memory devices. The USB I/O interface was selected because USB Version 1.1 can be implemented entirely in digital logic (with no analog components) and because USB is a popular

modern, high-speed, serial interface representative of the current trend toward high-speed serial I/O interfaces. A 48MHz oscillator chip was added in order to enable operation of the USB interface at 12Mbps. [4] provides Information on example Verilog and VHDL designs for a USB 1.1 protocol analyzer using this interface. SRAM and EPROM devices were added to enable experimentation with circuits requiring external memory, such as DSP and general-purpose microprocessor designs. In a class on "Computer Design" at our university, students were able to implement a pipelined RISC microprocessor for the ARM THUMB instruction set using this circuitry [4].

4. DISCUSSION

This paper has presented the design of a general-purpose FPGA prototyping board suitable for use in a university course on digital logic design, advanced digital logic design, or computer design. By using this type of FPGA prototyping board, students can learn to implement advanced digital designs, possibly involving memory devices or fast serial I/O interfaces, using state-of-the-art FPGA devices while avoiding the use of overly "canned" solutions.

7. REFERENCES

- [1] www.synopsys.com, home page for Synopsys Inc., a synthesis tool vendor.
- [2] www.exsedia.com, home page for Exsedia, an ASM-based high-level design entry and simulation tool vendor.
- [3] www.xilinx.com, home page for Xilinx Inc., an FPGA vendor.
- [4] S. Lee, *Advanced Digital Logic Design: State Machine Design Using VHDL, Verilog, and Synthesis for FPGAs*, Brooks/Cole Publishing, Belmont, 2004.